SAFETY CHARACTERISTICS IN SYSTEM APPLICATION SOFTWARE FOR HUMAN RATED EXPLORATION MISSIONS FOR THE 8TH IAASS CONFERENCE

Edward J. Mango [(1)]

[(1)] National Aeronautics and Space Administration, Kennedy Space Center, Florida, U.S.

ABSTRACT

NASA and its industry and international partners are embarking on a bold and inspiring development effort to design and build an exploration class space system. The space system is made up of the Orion system, the Space Launch System (SLS) and the Ground Systems Development and Operations (GSDO) system. All are highly coupled together and dependent on each other for the combined safety of the space system. A key area of system safety focus needs to be in the ground and flight application software system (GFAS). In the development, certification and operations of GFAS, there are a series of safety characteristics that define the approach to ensure mission success. This paper will explore and examine the safety characteristics of the GFAS development.

The GFAS system integrates the flight software packages of the Orion and SLS with the ground systems and launch countdown sequencers through the 'agile' software development process. A unique approach is needed to develop the GFAS project capabilities within this agile process. NASA has defined the software development process through a set of standards. The standards were written during the infancy of the so-called industry 'agile development' movement and must be tailored to adapt to the highly integrated environment of human exploration systems. Safety of the space systems and the eventual crew on board is paramount during the preparation of the exploration flight systems. A series of software safety characteristics have been incorporated into the development and certification efforts to ensure readiness for use and compatibility with the space systems.

Three underlining factors in the exploration architecture require the GFAS system to be unique in its approach to ensure safety for the space systems, both the flight as well as the ground systems. The first are the missions themselves, which are exploration in nature, and go far beyond the comfort of low Earth orbit operations. The second is the current exploration system will launch only one mission per year even less during its developmental phases. Finally, the third is the partnered approach through the use of many different prime contractors, including commercial and international partners, to design and build the exploration systems. These three factors make the challenges to meet the mission preparations and the safety expectations extremely difficult to implement.

As NASA leads a team of partners in the exploration beyond earth's influence, it is a safety imperative that the application software used to test, checkout, prepare and launch the exploration systems put safety of the hardware and mission first. Software safety characteristics are built into the design and development process to enable the human rated systems to begin their missions safely and successfully. Exploration missions beyond Earth are inherently risky, however, with solid safety approaches in both hardware and software, the boldness of these missions can be realized for all on the home planet.

EXPLORATION SYSTEMS STRUCTURE

NASA's Exploration Systems Development is building the agency's crew vehicle, next generation rocket, and ground systems and operations to enable human exploration throughout deep space — a capability the world has not had for more than 40 years.

The Orion spacecraft, Space Launch System (SLS) and a modernized Kennedy Space Center spaceport will support missions to multiple deep space destinations extending beyond our Moon, to Mars and across our solar system. This innovative approach aligns with NASA's bold new mission to design and build the capability to extend human existence to deep space.

A program at the Johnson Space Center manages the Orion, a program at the Marshal Space Flight Center manages the SLS and a program at Kennedy Space Center manages the Ground Systems Development and Operations (GSDO). The GSDO program is broken down into major projects under the program. Command, Control and Communications (C3) is one of those projects. The Ground and Flight Application Software, GFAS, is a project within GSDO.

EXPLORATION MISSIONS

Human exploration missions are in the critical development phases. All three programs under ESD have completed their various Critical Design Reviews (CDRs). Each of the CDRs is at different levels of maturity with the details of their designs. In order to be ready for human flight in the 2021 timeframe a series of two highly integrated flights, test flights prior to a human test flight are required. The first flight test was the Exploration Flight Test -1 (EFT-1) in December 2014. The Orion capsule was launched on a Delta 4 V launch vehicle. The goal was to test the capsule structure as well as re-entry thermal and guidance. The second flight test is Exploration Mission – 1, (EM-1). This first fully integrated mission of NASA's Orion spacecraft and Space Launch System (SLS) will launch from a modernized Kennedy spaceport. The 70-ton SLS will send an uncrewed Orion into lunar distant retrograde orbit, a large orbit around the moon that is farther into space than any human spaceflight system has ever ventured. The mission will last 25 days and splashdown in the Pacific Ocean. Exploration mission – 2 (EM-2) will be the first "crewed mission" of Orion launched from Kennedy spaceport again using the Space Launch System. The flight will focus on checking out the mission systems, crew systems, and demonstrating the capability for the crew to operate this vehicle in deep space, near the moon, and return safely to earth. The mission is planned to be two weeks in duration. After the EM-2 flight, the integrated systems will be used to perform all of the future human exploration missions beyond Earth. Destinations and complexity will evolve as the missions are defined and developed. The entire architecture is based on capability not only on destination.

GROUND AND FLIGHT APPLICATION SOFTWARE

The overall construct of the C3 project includes the Launch Control System (LCS) as well as the GFAS system. The LCS comprises a software and hardware platform structure and includes a set of system software that operates and interfaces with each of the flight vehicles, as well as the various ground systems' interfaces, and the Launch Control Center (LCC). Consoles within the LCC Firing Room are used by both the flight vehicle and ground system engineers to monitor and command the hardware, perform the testing, checkout, and preparations including launch countdown of the flight vehicles. Each of the flight hardware vehicles has its own avionics and flight software that must interface with the LCS. The Orion spacecraft, the SLS Core vehicle and the Interim Cryogenic Propulsion Stage (ICPS) all have different approaches to their corresponding flight avionics systems and flight software. LCS must interface with each of these three different interface approaches as well as integrate the downlinked information and command uplink information all through a common set of equipment for the various Firing Room engineers. The LCS platform system software is similar to various operating systems, like Windows or Apple OS operating systems.

As with most command and control architecture, there needs to be applications on top of the platform of software structure. The GFAS system is the equivalent command and control applications on those various operating systems. The GFAS applications integrate the flight software packages of the Orion, the flight software of the SLS, and the ground control systems through the LCS. The launch countdown sequencers are also in the GFAS applications. All GFAS applications use the 'agile' software development process.

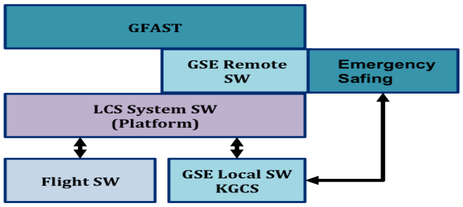

Figure 1 – GFAS Application Structure

The GFAS will develop the integrated firing room console applications and displays for pre and post launch activities to support flight and ground processing and integrated ground subsystem processing as required for Orion, Core Stage, Booster and ICPS. These software applications and displays will support integrated Ground Systems verification and validation, launch vehicle and spacecraft integration and launch processing. These displays and applications will also support off-line processing, recovery, and de-servicing of the Orion after splashdown.

The applications and displays include Graphical User Interface (GUI) and applications for Command Control Sequences, Prerequisite Logic, Reactive Control Logic, and Data Fusion.

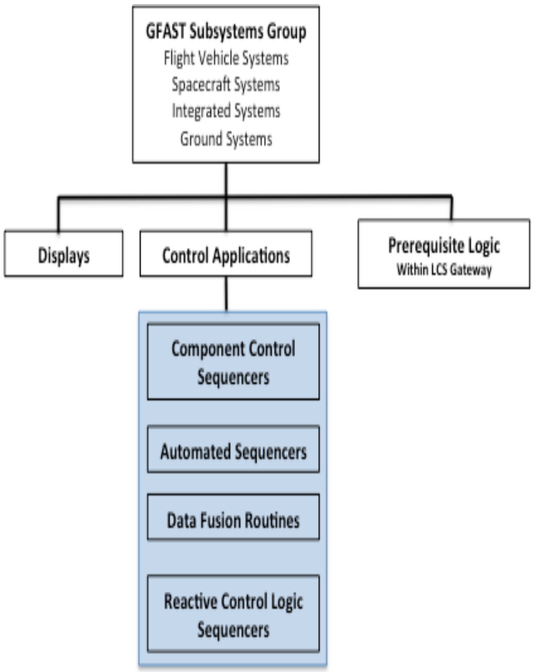

Figure 2 – GFAS Application Types

The GFAS software is used to satisfy integrated subsystem requirements, integrated ground to flight requirements, and integrated flight processing requirements. The GFAS team is made up of ten grouped subsystems comprising of approximately 70 software and processing engineers. Each subsystem team supports all the applications for the subsystem regardless of which flight vehicle they are supporting. For example, Avionics team supports the avionics installed in Orion, ICPS, SLS, Boosters and Engines. Below is the listing of each of the GFAS Teams.

Avionics
Cryo/Prop
Electrical
ECLSS/Hypers
Flt Controls
Mech
APU/HYD
Comm
Integration
Master

Figure 3 – GFAS Software Teams

The LCS system is developed through a series of Agile processes as well and is required to support Ground system verification and validation in late summer of 2016. The GFAS subsystems are expected to have verifications completed prior to first use of any applications used in flight hardware processing and testing. The first critical milestone for this effort is currently expected in Fall of 2017 against the Orion vehicle in the Multi-Purpose Processing Facility (MPPF). The Orion vehicle will go through a series of commodity loading activities and initial vehicle tests, prior to integration of the spacecraft hardware with the SLS in the Vertical Assembly Building (VAB).

SOFTWARE DEVELOPMENT PLAN

The GFAS project uses a form of "agile development" like many software application projects. Agile development is an alternative to classic project management techniques, in that it adapts quickly to unpredictable and multi-faceted requirement, design, verification and production situations. Agile is an alternative to a standard "waterfall" or traditional sequential project development scheme. For software applications, 'agile develop' is ideal when the vehicle hardware and interface hardware is still in design and development. Otherwise a classic project approach would require software applications to be developed after the hardware design is matured, qualified and even in production, thus causing an extensive and lengthy overall program development schedule. In the case of the ESD development, without agile development to many of the various software products, at least two or more years would be added to the overall program development, as the tasks would be sequential instead of paralleled. The Agile project development approach provides opportunity to assess and evaluate the direction of a project throughout the development lifecycle and adjust development priorities to match the assessment. This saves time during development. To achieve this project effort, work is accomplished via a series of cadences called 'sprints'. For GFAS, a series of 'sprints' are made up into a Drop. Each Drop has a specific set of actionable products to produce. Each 'sprint' within a Drop is two weeks in duration, with a total of six 'sprints' per twelve-week Drop. Thus the development can be measured in short increments. Changes, alterations, and issues that arise can be discussed and implemented within a Drop, and even from 'sprint' to 'sprint'. There are a total of fourteen Drops for the overall GFAS project development. In the Spring of 2016, GFAS will complete Drop 6 and start Drop 7. Additional Drops of requirement content, and software development activities can be added, if needed, due to flight hardware design modifications occur, or issues are found with ground or flight qualification activities. This approach allows the focus on safety and preparing the correct design for each mission.

The GFAS project control is through two separate activities. As with any project there are the three pillars of project management, cost, schedule and technical project management. The Technical Review Panel (TRP) is used to support the technical aspects of the GFAS project. The TRP, baselines or modifies software design requirements and implementation methods to include verification and validation. The TRP is the gateway between the driving NASA standards, specifications, and the flight/ground hardware design solutions and requirements. The TRP meets at least weekly to codify and accept the various GFAS team's detailed development plans. The second portion of the project control is the GFAS Drop Reviews. These Drop Reviews evaluate progress against the near term plan established for each twelve week Drop. Progress is measured in schedule performance, and work effort against the expectations for that Drop. The Drop Reviews, completed at the beginning, mid-point and end of each Drop help establish the cost and schedule impacts to the overall GFAS project effort.

The figure below illustrates the Agile process used within the GFAS project.

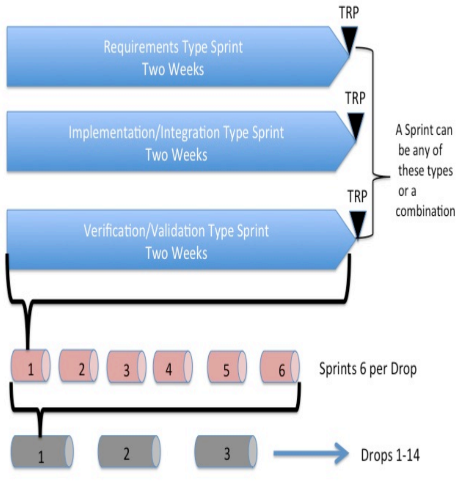
TRP
Requirements Type Sprint
Two Weeks
TRP
Implementation/Integration Type Sprint
Two Weeks
A Sprint can be any of these types or a combination
TRP
Verification/Validation Type Sprint
Two Weeks
1
2
3
4
5
6
Sprints 6 per Drop
1
2
3
Drops 1-14

Figure 4 – GFAST Agile Process

The application software design lifecycle is made up of six phases for each set of the ten application's teams. These lifecycles are broken down by subsystem for the teams processing site locations which require an application software set of deliverables. For example, the electrical applications used during the MPPF processing may be totally different than those used during the Vertical Assembly Building (VAB) processing. The six phases include the Doc 45 Design Review, the Doc 90 Design Review, Implementation, Integration, Verification and finally Validation. Release of the application products for production use occurs after each of the lifecycle phases is completed for the software required for the site deliverable. During any specific 'sprint' or Drop, different subsystems may have various application products in the different phases. Therefore, not every application is at the same state of development and it's that flexibility that makes the overall software development approach 'agile'.

Doc 45 Software Design Review

The preliminary design development is commonly called the Doc 45 Design Review. This is the initial development of the requirements for the applications for specific subsystems. Since during this phase, many of the flight hardware design states are unknown or still not mature enough in their designs, the content for the Doc 45 Design Review is considered preliminary. At this phase the Software Requirements and Design Specifications (SRDS) are started. The SDRS will address the scope of the software applications, the types of applications, either Displays, Sequencers and/or Algorithms. In addition, the SRDS includes the approximate number of measurements and commands the applications are expected to process, the operational scenarios for the applications and any interface requirements to other applications will be documented along with the LCS platform needs or the flight hardware software interfaces. Each SRDS is brought forward to the TRP for approval.

Doc 90 Software Design Review

At the Doc 90 Design Review the software design package is ready for final review and the start of implementation. At this point the flight hardware requirements are established, through the Operational Maintenance Requirements Set (OMRS), the Launch Commit Criteria (LCC), software standards and specifications and the flight hardware subsystem critical design. A mapping of the software applications against each of the established requirements is completed and the resulting product is called the Requirements Traceability Verification Matrix (RTVM). The RTVM is critical to establish the proper links between the flight hardware requirements, the safety specifications, and the detail software applications and code for implementation. The RTVM is used throughout the rest of the software lifecycle including Verification and Validation. From the RTVM, a set of Design Verification Objectives and Design Validation Objectives (DVO's and DVaO's) are produced and baseline for use during verification and validation phases. The software Doc 90 Design package is accepted by the TRP as well as by the independent technical authorities as a check and balance of the software design.

Implementation

Implementation is the next phase in the software development process. Implementation is the actual display development and software coding of the applications, scripts, sequencers and control logic. For GFAS, the applications are coded in a C++ based Application Control Language (ACL). During this phase the applications are put under configuration control. Individual applications or components of an application are tested for syntax errors, logic gates, logic interfaces and also for its compatibility to the RTVM. In order to remain as agile as possible, a certain amount of implementation can be worked in parallel with the Doc 45 and Doc 90 design effort. This is necessary in the GFAS development lifecycle, as the flight hardware design and development is not often at a state for the ground software applications to proceed with fully accepted design details. This approach is done at risk both to the schedule but more importantly to the technical content of the software requirements and actual code. The TRP process and the GFAS project management pays particular close attention to the risk of performing this type of agile development effort. Once the implementation phase is near completion, there is a re-alignment of the software applications to the RTVM and the corresponding DVO's and DVaO's. This is critical in the extremely agile environment of the GFAS project to ensure all the requirements for usability are met, and safety remains as a focus for implementation at each step of the lifecycle.

Integration

The Integration phase provides the initial integration of the individual software applications to be compiled into a configuration managed Test Configuration Identifier Description (TCID). The complied action is the first time the application is incorporated into the LCS platform. Integration testing includes checks to assure that the applications have the appropriate measurement identifier information, and can properly communicate with other portions of the LCS systems. Any regression testing due to changes in the LCS platform operating system could also be performed during this phase against the application. The DVO's and DVaO's are incorporated into a set of software verification test procedures. A review and alignment of the SRDS, the RTVM, the application, and the verification procedures are completed by the TRP and accepted for readiness to begin the next step of the lifecycle, verification.

Verification

The Software Verification phase contains the verification activities needed to provide software requirements buyoff from quality and safety for the GFAS Displays and Applications. This testing is performed to ensure that all allocated functional requirements, including DVOs, are adequately and correctly satisfied. A Software Test Readiness Review (STRR) is conducted early in the phase to ensure the GFAS software team and the project stakeholders are all cognizant of the test plans, procedures and schedule to complete the appropriate testing. Verification Test execution is performed by a member of the GFAS team that was not directly involved in the development of the software. This provides a level of independent verification. In addition, the Safety and Mission Assurance personnel buy-off the test procedure steps, as well as the closeout of the procedure. A formal non-conformance process is also in place during the testing. Any non-conformance is formally documented and appropriate corrective action is put in place before the GFAS application set is considered complete through the verification test process.

Validation

The Validation phase is the final phase prior to formal release of the software applications to the operational TCID set. Once the verification testing is completed, and any non-conformances are formally disposition, the applications are ready for validation testing. Like verification testing, most validation testing is performed against a set of software models built by a separate software organization. The models are developed to simulate the flight hardware configurations. Most are not full physics based models, but are adequate to verify the application software. For validation testing, the stakeholder must accept the software for their use. In the case of processing for Orion, ICPS and SLS, the stakeholders are the operations and processing engineers and test controllers who will be performing the various checkout tests and processing on the flight hardware. In order to help ensure stakeholder needs were fully incorporated into the GFAS applications, the GFAST project appointed the GFAS "leads" from the operations engineering community at KSC. So the validation testing will include members of the GFAS teams as well as other operations engineers. A few of the validation tests will also be performed against the flight hardware test rigs at the hardware integration test rigs in Denver and Huntsville. Validation testing is executed in a similar manner to the verification testing with complete configuration control, S&MA oversight and non-conformance processing. Once the Validation testing is complete, the GFAS Application set is ready to use in an operational TCID and ready for a flight hardware Test Readiness Review (TRR).

SOFTWARE SAFETY ANALYSIS

Software projects under NASA's control are subject to the NASA Standard NPR 7150.2B. This Standard addresses the various top-level standards for software development, safety and configuration management. Individual software projects derive lower level software requirements and standards from the set of standards found in the NPR. Under the GFAS project there are a number of additional standards, which must be followed for proper development. They include the End To End Command and Control Software Development Plan, as well as the GSDO End To End Command and Control System and Software Safety Plan, and End-to-End Command and Control Software Assurance Plan. Each of these is a lower level derived from the NPR 7150.2B.

There are a number of software safety characteristics built into the GFAS system. Throughout the six phases of the project development, software safety characteristics are embedded into the phases. Each of these characteristics adds a critical safety element in the overall structure of the GFAS software applications. Besides those characteristics embedded in the development phases, there are five overarching characteristics that help ensure GFAS is ready and capable of performing human spaceflight mission preparations

The first of these characteristics is the classification of GFAS software. Since GFAS software applications are used in the human spaceflight, GFAS must follow the elements of the Class A classification for NASA software according to NPR 7150.2B. The Class A classification is applicable to software systems and applications that are used to ensure safe and sustainable software for human rated space vehicles, during ground and flight operations. For GFAS, this represents all of the applications used to monitor, command and control the various elements of the flight hardware from initial power up at KSC through the completion of a launch, scrub recycle and recovery operations. The classification is applicable to all 1.5+ million lines of code, thousands of applications and displays across the ten GFAS subsystems.

Another key characteristic is the embedding of system safety engineers into the design and implementation process. The GFAS system safety engineers are tasked with ensuring that the subsystem software design is safe and reliable when used for its intended purpose. During the design and implementation of the flight programs, the flight system safety engineer is responsible for analyzing the overall subsystem design and the hardware component failures that can lead to subsystem level hazards. This is accomplished by performing a hazards analysis and a qualitative reliability analyses, or failure modes and effects analysis upon the subsystem design. If the analysis performed by the system safety engineer identifies concerns within the design, it is then the responsibility of the system safety engineer to raise the concerns through the flight system design review process. The GFAS software safety engineers review the SRDS both at the Doc 45 and the Doc 90 reviews for incorporation of the flight system hazards through the proper implementation of the OMRS into the software applications. Since the GFAS software is used to implement flight hardware activities and preparations for flight, many of the hazards and hazard analysis developed by the flight programs are translated into other operational requirements and are further implemented into the GFAS software code. Thus the system safety engineers who are assigned to the GFAS subsystem teams are focused on the proper implementation of those flight hardware hazards through the OMRS, the other operational requirements, into the software code.

Another characteristic is the review and concurrence the safety engineers perform on the software code itself to ensure the software applications meet the software standards as defined. These reviews ensure that applications for displays, command control sequences, prerequisite logic, reactive control logic, and data fusion are standardized. In addition, the software chief engineer must concur with the application design through the approval of any deviation against the software application standards.

Yet another overarching characteristic is the quality assurance effort required throughout the development process. Quality engineering supports the Doc 45 and Doc 90 reviews and once baselined, using the SRDS and the RVTM, the Quality Engineer ensures through each step in the development that the software application meets the RVTM. This is critical in the Integration, Verification and Validation steps in the development process. It is also critical that any non-conformance is addressed against the software design requirements (SRDS) and traceable back to the RTVM.

The final characteristic is the configuration control of the software application from its initial implementation through final validation and inclusion into the TCID used against flight hardware. There is a rigorous process that aligns with the standards defined in NPR 7150.2B. The GFAS team as a whole is accountable to configuration control of the software and the clear traceability from the requirements, through the implementation and into the verification and validation efforts. Ultimately, at the Design Certification Review (DCR), the GFAS project attests to the completeness and readiness of the GFAS applications to perform their intended purpose against the flight and ground hardware.

These five overarching characteristics are key to building a foundation of application software that will be safe to use against human rated flight hardware.

CONCLUSION

The ESD is in the midst of developing the first human rated beyond Earth space exploration capabilities in over 40 years. The capabilities being developed will bring human crews further into the exploration of our solar system than ever before. As the hardware and software capabilities are developed on the flight systems, a critical portion of the overall development is the design, development and implementation of ground applications to test, checkout, prepare and launch the flight systems. The GFAS project is made up of ten software subsystems and is closely aligned, to the flight hardware and software systems. The GFAS project follows a series of software requirements and standards. The implementation of these NASA Agency and GSDO Program requirements and standards is completed through a set of GFAS software safety characteristics to ensure safe and effective software applications. As the missions are designed, it is the solid safety approaches that will ensure the inherently risky missions can be successfully carried out for human exploration of our near space neighborhood.

Image - 1

REFERENCE

[1] 2014, National Aeronautics and Space Administration, *NASA Software Engineering Requirements,* NASA NPR 7150.2

[2] 2015, National Aeronautics and Space Administration, Ground Systems Development And Operations Program, *End To End Command And Control Software Development Plan, Volume 1,* K000064476-PLN Rev: B

[3] 2016, National Aeronautics and Space Administration, Ground Systems Development And Operations Program, *End to End GSE/GFAST Subsystem Implementation Standards,* K0000254500- GEN

[4] 2014, National Aeronautics and Space Administration, Ground Systems Development And Operations Program, *End-To-End Command and Control System and Software Safety Plan,* C3R-E2ECC-303 Rev: A

[5] 2015, National Aeronautics and Space Administration, Ground Systems Development And Operations Program, *End-To-End Command and Control System Software Assurance Plan,* K0000136629-PLN Rev: C

IMAGE

[1] 2016, National Aeronautics and Space Administration, *Space Launch System First Flight*